January/February 1986 Edition

RE RUN

RUN Programs on Disk

*For the C-64 and C-128**

Plus:
Bonus
Programs!

*128 mode programs included

Introduction

January-February '86 ReRUN

This issue of ReRUN is our first edition for 1986 and is one of our largest to date. All the programs that appeared in *RUN* magazine in January and February of 1986 are included, as well as many of the programs from the December 1985 issue and a never-before-published bonus program. For those of you who have been waiting for DFSTRUCTURE and DFRESTRUCTURE (from the DATAFILE series), here they are!

Directory titles are capitalized below and should be used for loading programs. Article titles, when they are different, appear in parentheses.

The new C-128 is represented with AUTOBOOT 128 (C-128 Autoboot Maker), 128 TYPIST (128 Perfect Typist) and ULTRA HIRES (128 Ultra Hi-Res Graphics). ULTRA HIRES is one program that you won't want to miss. It is the first program that really takes advantage of the C-128's 80-column mode for high-resolution graphics, which Commodore documentation led us to believe was not possible. We are including five demonstration programs for ULTRA HIRES so you can quickly see what your C-128 is capable of doing.

Several programming utilities are included on this issue. BASIC AID 64 will add Renumber, Find, Change and Reset commands to BASIC 2.0. AUTORUN 64 allows you to create self-running programs.

LISTER 64 (Add Elegance to Listings) is a handy utility that will format your listings for output to the screen or to a printer. These formatted listings hold indentations and spacing between lines, which make reading and debugging of programs much easier. DISK RENAMER 64 (New Names for Disks) is a disk utility that will let you rename your disks.

In the last few months, *RUN* magazine has implemented several new proofreader programs that make typing in listings easier. We are including our newest proofreader, 128 TYPIST (128 Perfect Typist) for C-128 mode Basic programs. We also have our new machine language entry program for the C-64, ML Typist (ML Perfect Typist). This will give you the ability to type in the hexadecimal code listings that we occasionally publish.

Productivity is a major thrust in home computing today, and *RUN* provides a wide range of applications for this purpose. RUNTERM PLUS is a sophisticated terminal program for telecommunicating with your C-64 that has received a tremendous response from our readers. It holds uploading and downloading capabilities as well as instant access to major on-line networks such as CompuServe, Delphi and The Source.

CREDIT CARDS 64 (Credit Card Keeper) makes it easier to keep track of your credit cards. It holds information on an unlimited number of cards. TAX DEDUCTION 64 (Tax Deductor's Delight) gives you a method of computerizing your deductions as tax time approaches.

DFRESTRUCTURE and DFSTRUCTURE are for use with Mike Konshak's popular DATAFILE series. With them, you can see the format of any file, change that format and get a printout of it.

Every ReRUN edition in 1986 will contain a never-before-published bonus program. This time, we give you BAROQUE MUSIC 64, a program that plays Johann Sebastian Bach's Invention #4. You are provided with music history anecdotes as this classic piece plays.

See you next time, when ReRUN will bring you *RUN*'s March and April programs.

Margaret Morabito
Technical Manager
RUN *magazine*

Directory

1	BASIC AID 64	C-64
4	AUTOBOOT 128	C-128 MODE
7	STACK 64	C-64
9	CREDIT CARDS 64	C-64
11	128 TYPIST	C-128 MODE
12	ML TYPIST 64	C-64
16	RUNTERM PLUS 64	C-64
25	WORD JUMBLER 64	C-64
27	PULSING PICS 64	C-64
29	AUTORUN 64	C-64
31	LISTER 64	C-64
34	DISK RENAMER 64	C-64
*36	ULTRA HIRES BOOT	C-128 MODE
36	ULTRA HIRES	C-128 MODE
36	PROGRAM BASE	C-128 MODE
36	DEMO.TEXT	C-128 MODE
36	DEMO.MOIRE	C-128 MODE
36	DEMO.3D BAR	C-128 MODE
36	DEMO.CIRCLE	C-128 MODE
36	DEMO.3D CUBE	C-128 MODE
46	TAX DEDUCTION 64	C-64
48	BURGERS 64	C-64
+50	DFRESTRUCTURE	C-64
+52	DFSTRUCTURE	C-64
£57	BAROQUE MUSIC 64	C-64

NOTE: All C-64 programs will work on the C-128 in 64 mode.
C-128 mode programs will ONLY work on the C-128 in 128 mode.

* ULTRA HIRES can only be used in 80 columns on the C-128.

\+ DFRESTRUCTURE and DFSTRUCTURE are two programs for use with Mike Konshak's DATAFILE program.

£ BONUS PROGRAM!

How To Load

C-64:

To load a C-64 program, type:

LOAD "PROGRAM NAME",8

and then press the RETURN key. The drive will whir while the screen prints LOADING, and then READY, with a flashing cursor beneath. Type RUN and press the RETURN key. The program will then begin.

Machine language programs will be loaded with LOAD "PROGRAM NAME",8,1. When these occur, they will be clearly marked for you on the directory page with an @ symbol.

C-128 in C-64 mode:

All C-64 programs can be run on the C-128 as long as your computer is in C-64 mode. This is accomplished by typing GO64 after powering up your C-128. Answer Y and press the RETURN key in response to the "Are you sure?" prompt. When in C-64 mode, use the instructions for loading C-64 programs above.

C-128 in C-128 mode:

Any C-128 mode programs are clearly labeled on the directory page. Your C-128 must be in C-128 mode to run these programs. All programs will work in both 40 and 80 columns, unless they are specifically marked otherwise.

To load a C-128 mode program, just press f2 and type the name of the program that you wish to load. Use the exact name that appears on the directory page. When the program has loaded, a READY prompt will appear. Beneath this is the flashing cursor where you should type RUN and press the RETURN key. The program will then run.

NOTES:

Before loading any program, always refer to the article in this booklet for special instructions.

It is a good idea to make a copy of your ReRUN disk before running the programs. You can accomplish this by using the disk backup programs that Commodore provides for the 1541 and 1571 disk drives.

64 BasicAid

By Robin Franzel

RUN It Right

C-64; C-128 (in C-64 mode)

The Commodore 64's full-screen editor makes it easy to write Basic programs. However, there are some features missing from the editor that would make you and your C-64 far more efficient. Even experienced programmers occasionally find the need to add more lines where there isn't any room. And the ability to find or change a string of characters makes debugging and corrections very easy!

BasicAid lets you add Renumber, Find, Change and Reset to your list of Basic commands. It resides in location $C400 (50176) and can operate concurrently with *RUN*'s previously published screen dump ("Print Your Screen," December 1984) and disk directory ("The Key to Your Disk Directory," August 1985) programs, as well as with the DOS wedge program.

RENUMBER COMMAND

To renumber program lines, including all the Goto, Gosub and If...Then branch targets and multiple branches (as in "ON A GOTO 100,200,300"), use the Renum command. The Renum command will start the program with line 10 and renumber by ten. To begin at another line number, you may enter that line number after the Renum command. The increment will remain ten unless you specify otherwise by adding a comma and the increment amount. For example,

```
RENUM 100,5
```

will make the first line of your program 100, the next line 105, then 110, and so on. (It takes approximately five seconds to renumber a 10K Basic program.)

FIND COMMAND

To find a string of data in your program, use the Find command. The text you wish to find must

be enclosed within delimiters. The delimiter may be any character. For example, to find each occurrence of the GOTO command, you could enter:

```
FIND /GOTO/
```

If you wanted to find GOTO in the program line

```
100 PRINT"GOTO THE STREAM"
```

the above command won't work. The search would be for the Goto command or token. To prevent the Basic interpreter from tokenizing your search string, simply use the quote mark as your delimiter. For example,

```
FIND"GOTO"
```

will find all the occurrences of the word GOTO that are not commands.

Each line containing the search string will be displayed on the screen. To prevent the lines from scrolling off the screen, the search stops after the eighth line, and the following message is displayed:

```
CONTINUE Y/N
```

Y will display the next eight lines of "hits," then the message again, while N will return you to Command mode.

CHANGE COMMAND

The Change command works like the Find command, except that the text you want to locate and the text you want to put in your program should be separated by your delimiter. Let's say you've mistakenly used the variable A$ for two different reasons, so you want to change some of the A$ variables to COPY$. Enter the following:

```
CHANGE /A$/COPY$/
```

The first line with A$ will be displayed followed by the message:

```
U=UPDATE  C=CONTINUE  S=STOP
```

U will cause the line to be changed and redisplayed, and it will display the next occurrence of A$; C will not cause the line to be changed, but will display the next line; and S will return you to Basic.

It's interesting to note the size limitation of the Find or Change strings. The buffer that holds the string is only ten characters long, but the length of the string is not necessarily limited to ten characters. For example,

```
FIND /PRINT"ENTER NUMBER"/
```

will trigger a String Too Long error, but

```
FIND /GOSUB10:GOSUB100:RETURN/
```

will not. This is because the Basic interpreter will convert Basic commands (unless within quotes) to a one-byte token, so the latter example is only ten bytes long!

RESET COMMAND

The only instance I can think of when you'd need to remove the BasicAid program from operation is when you want to load

a machine language program that uses the same memory space. In that case, the system will crash because the Basic vector will cause a jump to code that isn't set up to handle input. Therefore, you must restore the vector to its original value by entering the Reset command. This will disable the new commands; if you want to reenable them, enter SYS 50176. R

C-128 Autoboot Maker

By David Darus and Louis Wallace

RUN It Right

C-128 (in C-128 mode); 1571 disk drive

A new feature of the C-128 is its ability to autoboot programs on specially prepared disks that are in the drive when the computer is first turned on. This allows application software to load and run itself without the user being required to do anything except turn on the computer. You can also use this feature in your own programs to customize the computer on power-up. You may choose whatever color combination you like best. For example, you can define the programmable function keys to meet your requirements every time you turn on the computer.

To cause a program to autoboot, the program—Basic or machine language—must first be present on the disk. The key, however, is what the computer finds on track 1, sector 0. Whenever the computer is turned on, it reads track 1, sector 0, looking for a special data pattern in much the same way it does when a cartridge is plugged into the machine. If that information is present, the data at track 1, sector 0 is read into the computer at $0B00 (2816). As part of system initialization, that data, if present, is executed.

The format of the data at track 1, sector 0 is very specific. The first three bytes must be the letters CBM, followed by CHR$(0) repeated four times. Next is an optional booting message, which can be up to 16 characters long, followed by two more CHR$(0)s.

After that are machine language instructions that point to the address of the Run or Boot command and execute that command. The address used is in low-byte/high-byte form and is calculated by adding to 2186 the length of the booting message plus 15. The low byte is that ad-

dress combined with 255 by the And command; the high byte is the calculated address divided by 256. The machine language instructions are CHR$(162), CHR$(low byte), CHR$(160), CHR$(high byte), CHR$(76), CHR$(165) and CHR$(174).

Next is the string RUN "filename" or BOOT"filename," followed by CHR$(0). The Run command is used if a Basic program is loaded and run, and BOOT is used if a machine language program is loaded and executed.

AUTOBOOT MAKER

Since this can be quite a tedious task every time you want to make an autoboot disk, we have written an autoboot maker utility program for the C-128.

Autoboot Maker effortlessly writes that special sequence of data to track 1, sector 0. The program asks you for a booting

message as well as the name of the program to autoboot. It also asks if the program is Basic or machine language and sets up the disk accordingly.

To protect you from overwriting a previously created autoboot disk, the program checks to see if the special data sequence is already present. If so, you are given the option to quit or to rewrite the autoboot code. Furthermore, to protect you from overwriting the autoboot sequence, the BAM (block allocation map) is updated to inform the disk drive that all of track 1 is being used, and it should not write any new data there. This results in the loss of 20 disk blocks (out of 1328). However, if you use the Collect command on an autoboot disk, it will make track 1 available for data storage, and the autoboot code could be destroyed.

When you run Autoboot Maker in 40-column mode, the screen will go blank for a while, as the C-128 goes into Fast mode and draws the graphics screen. After about 20 seconds, the screen is saved to disk under the names AUTO.PIC and AUTO.COL. On subsequent runs, the graphics screen will load up directly from disk. On the 1571, this takes about three seconds to load. In 80 columns, the graphics screens are bypassed.

You are then prompted to insert a disk to which the autoboot code is to be written. You must already have saved onto your disk a 128-mode program that you want autobooted *before* you run Autoboot Maker. You will be prompted for all needed information, and the customized autoboot code will be quickly written to your disk.

Autobooting disks is one of the ways the C-128 has made disk control user-friendly. Use it to customize the computer to your specifications. Now, all you'll have to do is place your specially prepared Autoboot-made disk into the 1571 drive and then turn on your C-128. Your program will automatically load and run. This would be particularly useful if you had a menu or a disk directory program that was autobooted. R

Stack

By Glenn W. Zuch

RUN It Right

C-64; C-128 (in C-64 mode)

This game of mental skill and logic is based on the popular Towers of Hanoi game. The object of this computer version is to move several bars from one pile to another pile, using a third pile as an intermediary. In moving them, you cannot place a longer bar on top of a shorter one. It sounds simple enough, but there are distinct rules you must follow if you expect to receive a reward for stacking the bars in the fewest possible number of moves.

You will find that this is a very user-friendly game. Everything you are required to do is clearly indicated on the screen. Every mistake you make brings up a prompt to remind you how to get back on the right track. If you follow the directions, none of your errors will count against your final score. You can even become a winner in more than one way.

Finally, you can choose any one of four levels of difficulty, moving up to the most difficult as your skill improves.

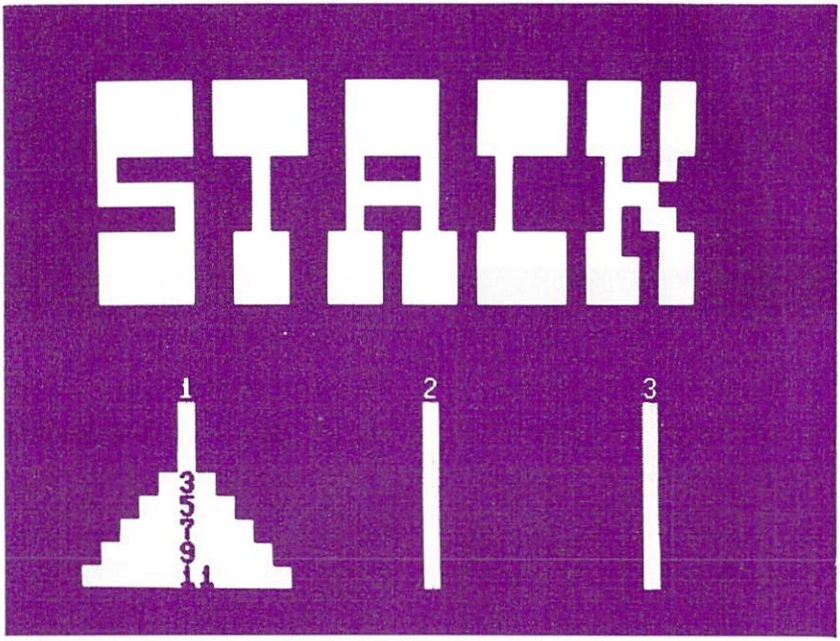
PLAYING THE GAME

When you run this game, you are first greeted by a title in large block letters and an illustration of the beginning stack of bars, each with its own identifying number. This is followed by some simple directions.

You are then given the option of choosing to move from two to five bars. The fewer bars, the easier it will be to solve the game. After you make your choice, the computer will display the starting setup, based on your choice, and tell you which bars you will be moving.

You are then asked which bar you want to move and where you want to move it. You move one bar at a time.

While the computer gives you directions and prompts you when you try to make incorrect entries, you must still make all of the decisions and computations to win. This, as you will soon discover, is not an easy task.



Eventually, you will succeed in stacking all the bars in the rightmost pile. The program will acknowledge if you are able to do this in the fewest possible number of moves. Even if it took you more than the minimum number of moves, you are still a winner! The computer will tell you how many moves you took and the least number of moves required, so that you will have a target to shoot for the next time you play. R

Credit Card Keeper

By Michael Reich

RUN It Right

C-64; C-128 (in C-64 mode)

Credit Card File uses Data statements to hold five pieces of information on each credit card: category, name of card, account number, expiration date and address of issuer.

The program is set up for three categories of credit cards: Gasoline, Stores and Other. You may change these categories to meet your particular needs. For each record (account), the five data items are entered beginning at line 1000. I separated the categories to make it easier to enter the information, but the program will work just as well if you mix them up.

One thing you must keep clear is the *order* of the Data statements for each record. From lines 300–370, the program reads all the data in the order listed above, so the Data statements *must* be in that same order.

Also, if you are missing a piece of information, such as the issuer's address, you must provide a temporary Data statement to hold its place; just create a Data statement with blank spaces. The last Data statement must be the word END, or you'll get an Out of Data error.

I allowed up to 18 letters for each card name, up to 19 characters for the account number, up to five characters for the expiration date (this usually consists of just the month and year—03/88, for example) and up to 39 characters, in two lines, for the issuer's address. You may change the size of these entries, but I suggest you keep the card names the same length, the account numbers the same, and so on.

MENU SELECTIONS

The first menu offers choices of reviewing the files, updating the data or quitting (selecting the Update option simply lists the program beginning at line 1000). Whenever you add or change

Data statements, remember to resave the program.

The second menu provides a variety of ways to view the files. You can look at each category separately or at the combined records for all categories. Whichever you choose, the program will show you the name, number, expiration date and address for each record, one at a time.

COMMAND LINE

During the display of the records, a command line will be displayed at the bottom of the screen. This line shows how to move through the lists of credit cards by pressing one of three keys: N for the next record in the category you selected; P for the prior record; and M to return to the menu. R

128 Perfect Typist

By James E. Borden

RUN It Right

C-128 (in C-128 mode)

128 Perfect Typist is similar to the 64 Perfect Typist that was first published in *RUN*'s September issue. If you've used 64 Perfect Typist, you'll know how to use 128 Perfect Typist.

Be sure your C-128 is in 128 mode before using 128 Perfect Typist. When you run it, you'll see the start-up screen. 128 Perfect Typist will always have the same SYS addresses: 5120 for on and 5150 for off. The only way to disable 128 Perfect Typist is with the SYS 5150 (off) command (or Reset, but that's a little drastic).

The 128 Perfect Typist will work on either a 40- or 80-column screen. Also, it lets you use the C-128's automatic line numbering. If Auto is on, the checksum will be printed below the line just entered, and the C-128 will print the next line number below the checksum. Although this allows fewer lines on the screen, they'll appear automatically. R

ML Perfect Typist For the C-64

By Robert Sims

RUN It Right

C-64; C-128 (in C-64 mode); disk drive

ML Perfect Typist is a checksum for machine language programs that you'll be typing in from *RUN* magazine on the C-64 (or the C-128 in C-64 mode).

When you load and run ML Perfect Typist, the Basic portion of the program Pokes into memory the machine code from the Data statements and executes the code.

TYPING IN MACHINE LANGUAGE LISTINGS

Machine language listings in *RUN* consist of a series of program lines, each of which begins with a four-digit number, the line number, followed by 13 pairs of characters, separated by spaces.

You do not enter the line number; the program enters it for you automatically. The spaces in the lines are for ease of reading only; they also appear automatically.

The pairs of characters are hexadecimal numbers. The first eleven pairs are the program's machine code. The last two pairs in each line are checksum values used by ML Perfect Typist to ensure that you've entered the line correctly.

To type in a machine language listing, first load and run ML Perfect Typist. At the opening menu, select 1 to enter a new program.

You will see a line number on the screen. Find this line number in the listing and type in the characters that follow the line number. ML Perfect Typist automatically takes care of spacing on the screen. If you make a mistake, simply delete your error and retype the character.

When you've typed in the 13 pairs of characters, compare what you see on the screen with the line in the magazine. If they're the

same, press the return key. If you find an error, use the delete key to back up to the incorrect character. Type the correct character, then reenter the rest of the line. Check it again, then press the return key.

If the line has been entered correctly, ML Perfect Typist will prompt you with the number of the next line to be entered. If you've made a mistake, Perfect Typist will alert you to this and prompt you to enter the same line again.

The line is not processed until you press the return key and ML Perfect Typist verifies the contents. The line is then converted to binary numbers and stored in its proper place in memory.

SAFETY FEATURES

ML Perfect Typist will accept only the keys 0–9, A–F, delete and return. If, for example, you mean to press the 7 key but hit the T key instead, ML Perfect Typist will ignore the T and wait for you to press a valid key.

In case of power failure or other disaster, it's a very good precaution to save your entries to disk every 50 lines or so.

COMMAND CODES

There are three command codes, generated with the CTRL key: CTRL-A, CTRL-C and CTRL-E.

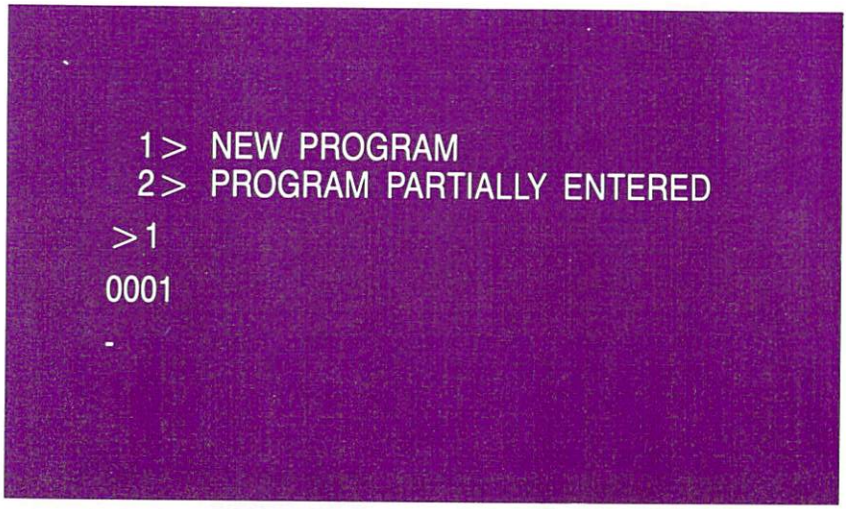
If you enter CTRL-A (hold down the CTRL key and press A), the program will abort. Only use CTRL-A if you become hopelessly confused while entering a listing. CTRL-A causes ML Perfect Typist to end without saving the entered lines to disk.

Use CTRL-C to cancel entry of a line. When you type CTRL-C, ML Perfect Typist will erase the line and prompt you to enter it again. Only the current line is affected.

HANDLING INCOMPLETE PROGRAMS

CTRL-E has two uses. First, use it if you want to end a session before you've entered the complete listing. Be sure, of course, to have a disk in the drive, or you'll lose the data you've already entered. Instead of entering the next line, press CTRL-E. You'll be asked for a filename. Then a short menu will be displayed. Choose option 2. When you select option 2, ML Perfect Typist will save the incomplete file to disk.

Caution: You *must* use option 2 to save your incomplete file! If you use option 1, you'll never be able to recover your program for further line entry. Also, if you name a file that exists on the disk, ML Perfect Typist will scratch the old file and replace it with the new one.



Later, when you're ready to resume entering the listing, load and run ML Perfect Typist and select 2 from the opening menu. You'll then receive a prompt for the filename. Enter the name of the incomplete file and press the return key. ML Perfect Typist will load that file into memory and then display your next line number. This is a handy feature because it always identifies the next line to be entered. Resume entry by typing in the character pairs in the line whose number is displayed.

STORING A COMPLETED PROGRAM

The second use for CTRL-E comes when you have entered the last line of a listing. When you are prompted with a line number larger than the last line number in the listing, enter CTRL-E and supply the filename. Next, you will see the menu. Select option 1 for storing a completed program, and ML Perfect Typist will store it to disk. Again, you *must* have a disk in the drive before you enter CTRL-E.

CAUTION!

If you follow instructions, ML Perfect Typist will give you an error-free copy of the machine language program you enter. However, keeping in mind the theory that, if anything can go wrong, it will, take precautions. When entering your program lines be sure that you are entering them in the correct sequence. Do this by checking the

line number on the screen against the one you are entering from the magazine.

Also, you will notice that all lines, except the last one, are the same length. Therefore, when you have entered all the characters in any line except the last one, your cursor should be in the 40th column on the screen. If it isn't, you must have left out one or more characters. If you press the return key at this point, you should get an error message and a prompt to retype the line.

ML Perfect Typist uses the last two character pairs of each line you enter as a checksum, to see whether you've entered the line correctly. There is a slight chance that, if you leave characters out of a line, ML Perfect Typist will add up the value of the characters, compare them to the last two character pairs, find a match and store an incorrect line. To avoid this, always check that your cursor is in the 40th position before you press the return key.

For the special case of the last line, just check to be sure you've entered the correct number of characters.

If you enter too many characters in a line, ML Perfect Typist will alert you to this and prompt you to enter the same line again. R

RUN's Great Communicator—Runterm Plus

By Robert Sims

RUN It Right

C-64; C-128 (in C-64 mode); modem

Runterm Plus is a full-featured terminal program that will communicate with any computer that uses either standard ASCII or Commodore ASCII data format. The program can transfer files by capture buffer, as well as by automatic protocols, which ensure error-free transmission. It also has one feature that lets you view high-resolution graphics and another that lets you play games through your modem.

This program is set up to work with the VIC-Modem, the 1650 Auto-Modem and other modems that are compatible with these.

Runterm Plus includes automatic file-transfer routines and modem game features, as well as the standard ASCII and full-duplex capability that you'll need to call the national networks. You will be able to upload and download program files with this terminal.

To use Runterm Plus, insert the disk into your drive. If you are using the VIC-Modem, set it for Originate. For the Commodore 1650 Auto-Modem or compatibles, set the modem for Originate and Telephone (Voice). Then enter:

```
LOAD"RUNTERM PLUS",8
```

and press the return key. When the load is finished, type RUN and hit the return key; you'll see the opening screen. Pressing CMD-M will display the menu of features available. When you're ready to go on-line, switch the modem (except the VIC-Modem) to Data.

CONTROL CODES

National services and some local bulletin boards require that

you enter a log-on code, which tells the host computer that you are ready to go. CompuServe, for example, waits for a Control-C (ASCII3) or a Return. If you are calling Delphi via Tymnet, you'll be asked to enter your terminal identifier (you type A).

In addition to the log-on code, you will need to send and receive other special control codes. The most common codes are composed of the first 27 characters in the ASCII character set—Control-A through Control-Z and Escape.

To generate these codes with Runterm Plus, hold down the CTRL key and press the appropriate key. CTRL-A sends a binary 1, CTRL-T (or the delete key) sends a binary 20, and so on. To send the escape character, hold down the CTRL and press the colon key. (ESC is often used on CompuServe.)

You should also be aware that if you press the home key, a CTRL-S (pause) is sent to the other computer. If you hit this key accidentally, then you should send a CTRL-Q (press the cursor-down key) to resume transmission. Pressing the run/stop key will send a CTRL-C, which is used as a cancel code on many systems.

OTHER SPECIAL KEYS

In the process of communicating with another computer, you will need to perform several auxiliary operations, such as checking the disk directory, capturing data in a buffer and saving it to disk, or preparing to receive a file. These operations are performed using local commands, generated via a combination of the Commodore logo key and letter keys.

For example, you can read the disk directory by holding down the Commodore logo key and pressing the D key. As it does for all local commands, Runterm Plus sends a CTRL-S (ASCII 19) to the other computer; this puts it on hold, so incoming data won't get mixed into the directory contents. The program then retrieves the disk directory and displays it to the screen.

When the last byte of the directory is displayed, Runterm Plus sends the other computer a CTRL-Q, telling it to resume transmission. All this is done automatically, so you needn't worry about it; I'm telling you this for your information only.

There are two more disk-maintenance commands—Logo-N and Logo-R. Use Logo-N to scratch a disk file. Simply type in the name of the file to be scratched, and Runterm Plus will remove it from the directory. Logo-R is used to rename a file. At the filename prompt, enter the change in this format:

```
newname = oldname
```

and hit the return key.

To see a menu of these local commands and the keys that generate them, hold down the Commodore logo key and Press M when Runterm Plus is running.

BUFFER COMMANDS

The Logo-O combination opens the 29K capture buffer; Logo-C closes it; and Logo-Z resets the pointers to the beginning of the buffer.

To capture characters, you must first open the buffer by pressing Logo-O. When it's full (about 117 blocks), you will see a Buffer Full message mixed in with the incoming characters. Any further data that's transferred will appear on the screen, but will not be captured.

Logo-B allows you to view the buffer contents on screen, and Logo-P will send the buffer contents to the printer, stripping out screen codes and control codes that your printer can't handle. You can abort either of these operations by pressing CTRL-X. (It may take the program a few seconds to acknowledge your command and stop the operation.) For faster results, hold down both the CTRL key and X until the abort is accomplished.

Two commands allow you to save the buffer contents to disk. Logo-U will save into a disk file all characters in the buffer. The program will ask you to supply the filename and filetype (program or sequential). Logo-S also saves buffer contents, but edits out screen codes and control codes so the saved file can be printed later.

The program handles all characters as Commodore ASCII. If you are calling a standard ASCII database, characters are translated to standard ASCII as they are sent. Incoming characters are converted to Commodore ASCII before they are stored. This means that all text is in a format compatible with your computer, so you don't have to concern yourself with conversions.

FILE AND BUFFER TRANSFERS

There are three ways of transferring files with Runterm Plus. The first involves loading the file from disk to the program's buffer area and uploading the data from there.

Second, using the XModem protocol, you can transfer files directly to and from your disk. Again, there is no need for conversion; programs transferred by XModem are ready to run when the transfer is done.

The third means of file transfer is via the Bozart transfer protocols.

If you wish to upload via the buffer, use the Logo-L command to load the file into the buffer—it will be loaded over anything that was there before.

To transfer the buffer contents to the other computer, you have two choices: Logo-V or Logo-Y. Logo-V will send the entire buffer contents without pause.

Logo-Y will ask you to enter a prompt character. The routine will then upload each line of the buffer, pausing after it sends each carriage return. During the pause, Runterm Plus examines incoming characters for the designated prompt character, which signals that the other computer is ready to receive the next line.

This transfer method is specially designed so you can upload messages from your disk to bulletin board systems. Generally, you can compose a message on your word processor and store it to disk as a CBM ASCII text file. Then, when you're on-line and want to send your message to the bulletin board, load the file using Logo-L and, when the BBS tells you to enter your message, press Logo-Y and supply the prompt character. Runterm Plus will then send the message contents automatically.

To use Logo-Y, you must, of course, know the prompt character being sent by the other computer. In most cases, this prompt character will be the last character in a menu or a start-of-line marker. Before using the Logo-Y command, try to notice which prompt character is being used.

On CompuServe, the prompt is usually a greater-than (>) or a colon (:). On Delphi, the prompt is usually a linefeed (you type CTRL-J when asked to supply the prompt) sent after a carriage return. Other systems simply send the return without a linefeed. If you don't see an obvious prompt character, try the linefeed or carriage return. Better yet, check the service's documentation or ask the SYSOP.

When you initiate either of these uploads, you'll be told that the upload is in process. When the upload is finished, the cursor will return and you'll be told that the transfer is complete. You can abort these uploads by entering CTRL-X.

SCREEN COMMANDS AND WORD WRAP

Three commands—Logo-H, Logo-I and Logo-J—change the border, background and text colors, respectively. If you are using Commodore ASCII, then you can also use the regular key combinations to change text color, just as you do when the C-64 is in Immediate mode.

Since the C-64 has a 40-column screen, and most telecommunication services use an 80-column format, you'll often find that incoming data will wrap around the screen, leaving the first part of a word dangling on the end of the previous screen line. If this irritates you or makes

the words hard to read, then you need Logo-W.

When Runterm Plus boots up, the Logo-W routine is set to eliminate word wrap. The screen is formatted for 40 columns. Broken words are erased from the previous line and moved to the next line for ease of reading.

However, there are times—typing in messages, for example—when you want to use 80-column format to keep up with spacing and the number of characters in a line. To turn off the justification routine, just type Logo-W. Each time you do this, the program will toggle and tell you its current status.

Some services—Delphi is one—have automatic word wrap. When using such services, you may want to set Runterm Plus to 80-column format (allow word wrap) so that the two justification features will not work against each other.

GRAPHICS AND LOGO COMMANDS

While the C-64's function keys are handy, there aren't enough of them. That's why Runterm Plus uses the logo key command format. As long as you're communicating with another computer that's using standard ASCII, there's no conflict. If, however, you are using Commodore ASCII and want to send the graphics characters represented by the logo and letter key combination, you must use the f8 key to leave Command mode and enter Graphics mode.

The f8 key is a toggle that switches the program between two states. When Runterm Plus is booted, it is in Command mode. This means you can use the logo key to generate local commands. If you are using Commodore ASCII and wish to send graphics characters, then just hit f8 to go into Graphics mode: hit it again when you need to use commands. Each time you toggle with f8, the program tells you which mode you have selected.

Logo-Q is used to end the program cleanly. If you hit it by mistake, the program allows you to change your mind.

THE OPENING MENU

You can configure Runterm Plus to fit almost any telecommunications format. When you load and run Runterm Plus, you will see an opening screen that gives you six options. You may set the program to call a specific national service, such as CompuServe, Delphi or The Source. If you are calling a BBS, you may choose between standard ASCII or Commodore ASCII. A custom terminal setting is also provided.

Runterm Plus will set up your

computer and modem to communicate with the specific service you have selected; you don't have to worry about stop bits, word length or other technical aspects of telecommunications. When using Runterm Plus, be sure to set your modem to the Originate mode. If you have a modem with a data/telephone (voice) switch, first make the call, listen for the carrier signal, then move the switch to data.

The selection of any option, except #6, will cause the screen to clear, signifying that Runterm Plus is ready.

If you choose option 6, you'll be asked to specify the parity, word length, stop bits, duplex and ASCII settings and the deletion character. Once you set these parameters, the screen will clear and you'll be ready to go on-line.

XMODEM TRANSFERS

To transfer a file directly to or from your disk, use the Logo-X command. The sequence for an XModem transfer is as follows.

You select download or upload from the other computer's menus. The other computer will prompt you when it's ready. You then press Logo-X, select X for XModem and type U or D.

If you select U for upload, Runterm Plus will ask you to supply the name of the existing file to be uploaded. You will be told when the transfer is completed, and keyboard control will resume.

If you select D for download, you will be asked for the filename and filetype. (You must not use an existing filename.) The program then creates the new file on the disk. The downloading process then becomes automatic. You need only wait until the program notifies you that the transfer is complete.

If noise on the phone line or some other problem causes the transmission to become garbled, then the garbled portion will be retransmitted to you. If Runterm Plus and the other computer are unable to complete the transfer, Runterm Plus will abort the transfer and return keyboard control to your computer.

A note about CompuServe: Using the XModem feature of Runterm Plus, you can download all files in CompuServe's Commodore Information Network. The special handshakes required to download programs with the extension .IMG are handled automatically by Runterm Plus.

HIGH-RESOLUTION GRAPHICS SCREENS

Runterm Plus can transfer and display high-resolution graphics screens. To view the screen while you're downloading it, it must be in what is commonly called Doodle format. That is, the file must be constructed so that the high-

resolution color area is placed first in the file, followed by a bit-mapped screen.

Although the file will transfer without problems, this hi-res feature will not give you a true copy of a multi-colored high-resolution screen, which is longer and in a different format than the regular bit-mapped two-color screen.

There are two ways to use this feature. You can either send a hi-res screen or receive one. Keep in mind, however, that your buffer contents will be overwritten during this process. Before viewing a screen, save your buffer contents to disk or print them out. If your local bulletin board uses Bozart protocols, you will be able to do graphics downloads from the board, too. Check with your SYSOP.

When Runterm Plus is booted, the hi-res feature is turned off. To activate this feature and view a hi-res screen while downloading, use the Logo-A command before you begin the downloading process. You can toggle on and off the Hi-Res Graphics mode by typing Logo-A.

To transfer a hi-res screen, first type Logo-A, as previously discussed, then, when the other computer is ready, select Logo-X to begin the transfer. Select Bozart protocol and then D for download, and the rest is automatic.

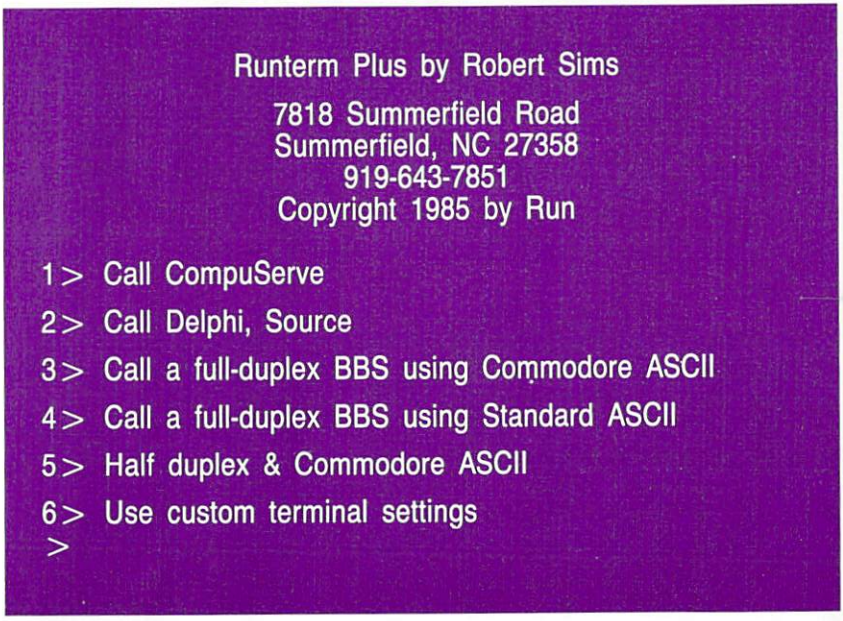
The special Bozart graphics protocol is similar to XModem, but faster, more accurate and more automatic. The protocol switches your computer to Hi-Res mode and back to regular screen mode. Also, it includes a handshaking feature, which checks to make sure that the incoming file will fit on your disk. (If the file won't fit, Runterm Plus will abort the file transfer and return to Terminal mode.)

In viewing hi-res files, you may experience a few seconds' delay while the two programs make the initial connection before transferring the file. During this delay, you'll be looking at the bit map of whatever was in the capture buffer. As the transfer progresses, you will see the background color change; then, while the file is being stored to disk, the bit-map screen will appear.

When the transfer is complete, Runterm Plus will pause for three seconds, to allow you to view the hi-res screen, then the screen will clear and you'll be back in Terminal mode.

Keep in mind that the Bozart protocol can only be used to swap files with another terminal program that uses Bozart protocol. Just as you can't speak English and expect to be understood by someone who speaks, say, only French, you can't expect two different transfer protocols to communicate.

Bozart protocol can be used to get an error-free transfer of



any file, not just graphics files. (The only difference is that you don't use Logo-A to view the non-graphics screen; you go directly to Logo-X for the transfer.)

Although the speed of transfer depends on the file being exchanged and, especially, on the number of repeated characters in the file, as a general rule you can expect to transfer a file 20 percent faster using the Bozart graphics protocols instead of XModem. For example, a typical 9000-byte graphics screen that takes about ten minutes to transfer with XModem will transfer in about seven minutes with Bozart protocol. Faster speeds are not possible, due to the constraints of the 300 bits-per-second speed of the modem.

An interesting side effect of this hi-res capability is that you can get a bit-mapped representation of any non-graphics file being transferred. Use Logo-A before the transfer, just as you would if the file were a graphics screen.

As the file is transferred, you'll see a kind of abstract design being built on your screen. Only actual hi-res screens will show you an organized picture. However, the abstract designs generated by a non-graphics file are

more interesting to watch than the constant waiting/transferring messages that normally accompany a file transfer.

CHANGING TERMINAL SETTINGS

There are times when you'll want to change your terminal settings without restarting the program. To redefine your delete key, use Logo-K. The standard delete character is an ASCII 127. However, some services require you to use a backspace (ASCII 8), and you may need to revert to the Commodore delete character, ASCII 20.

Depending on which service you select from the opening menu, Runterm Plus sets the delete character. For CompuServe and for linkups using Commodore ASCII, the setting is the regular ASCII 20; for Delphi and The Source, the setting is ASCII 127 (true delete).

To reset any or all of your terminal parameters, use Logo-T. This command will review each setting and prompt you to reset it.

MODEM GAMES

The Logo-G command allows you to load and run a special game program while on-line. This feature lets two people with copies of Runterm Plus play a game over their modems during a regular telecommunications session, without breaking the connection or switching programs.

To use this feature, both parties must have a game written specifically for Runterm Plus. Ordinary games will not work, because they load into the same locations in memory where Runterm Plus code resides.

Runterm Plus is designed to accommodate games that use the normal screen and keyboard cursor controls, as well as games that use multiple high-resolution screens and very fast action. If high-resolution graphics are not used in a game, Runterm Plus will accommodate a game program up to 30K in length. With hi-res screens, Runterm Plus will accept a game program of up to 20K, while leaving some memory still available. R

A-Maze-ing Word Jumbler

By Penny DeGroff

RUN It Right

C-64; C-128 (in C-64 mode)

In most computer word-scramble games, the computer scrambles a word and displays it on the screen. The player then types in the correct word.

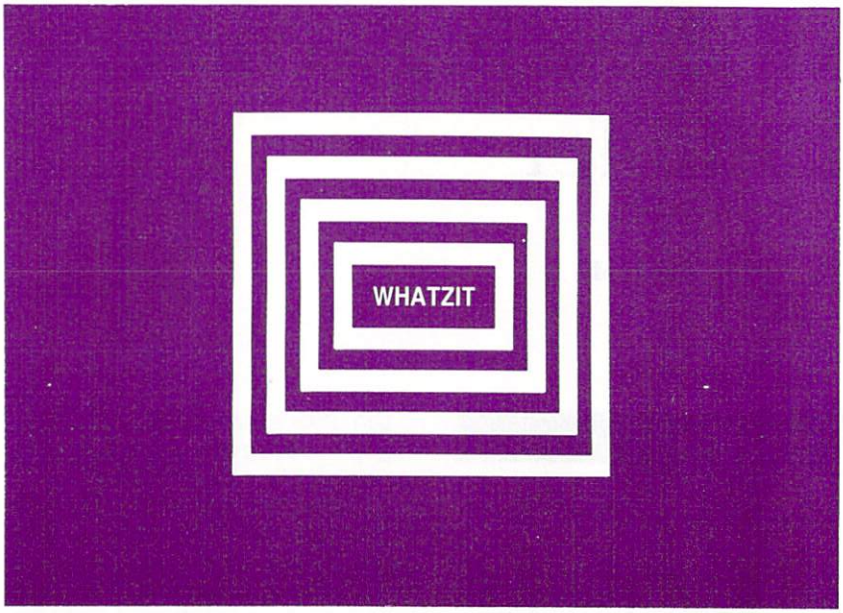
This game, Word Jumbler, is a little different. First, you are presented with eight categories, each dealing with a different subject. With a joystick in port 2, you move a white cursor beside the category you want, then press the fire-button to begin the game.

The playfield, a 12-section maze, will appear on the screen. Some sections will contain a letter; one section will contain a star; the rest will be empty. The letters, when combined in the correct order, will spell a word that relates to the category chosen. (For instance, the "I'm Hungry" category contains the names of various foods.)

To display the letters, one at a time, in their proper order, use the joystick to move the star directly under each letter and press the fire-button. That letter will then appear at the top of the screen. Continue this until all letters are removed from the maze, displaying a word.

At the bottom of the screen is the score section. When the maze is cleared, the word being played is shown along with the score and any bonus points earned.

If you've spelled the word correctly, you will earn one point for each letter in the word. There is no time limit for clearing the maze. However, bonus points are awarded if you do this in less than 30 seconds; you gain one point for each of the remaining seconds.



Since the star must go around the letters instead of over them, you can clear the screen more quickly by going through the empty sections whenever possible.

Play continues until five words have been unscrambled. The grand total and high score are then shown. The option of playing another game is also presented at this time.

MAKING CHANGES

Any or all of the categories or words within them may be changed. Lines 160–200 contain the eight categories currently used. There are 20 words in each category. They are contained in Data statements beginning at line 950. Since the maze contains 12 sections, no word can have more than 11 letters. Have fun! R

Pulsing Pictures

By George Trepal

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive

Unlike conventional keyboard graphics programs, this program produces movement, so you can have waterfalls, moving belts, wheels, flashing patterns and many other effects. You can also use this program for advertising and making special announcements.

If you haven't used keyboard graphics before, look at your computer's keyboard. Most keys have two little pictures, or graphics, on them. To print a key's left-most graphic, hold down the Commodore logo key (bottom left of keyboard) and press the desired key. To print a key's right-most graphic, hold down the shift key and press the desired key. To change to color, hold down the CTRL key and press the color key (upper row) of your choice. For other colors, press the Commodore logo key and a color key. Key 1 is orange, 2 is brown, 3 is light red, 4 is dark gray, 5 is medium gray, 6 is light green, 7 is light blue and 8 is light gray.

You can combine the little pictures to make drawings, borders and other things. You can also put text on the screen. The very bottom right-hand corner is forbidden territory. If you print a character here, the screen will scroll up and you'll lose the top line of your picture.

When you run the program, the screen will clear. The keyboard graphics work as usual, except for the numbers and some punctuation marks. The program will explain which characters are different when you run it. Pressing any of these altered keys will put a moving character on the screen. These characters can be combined with letters and normal keyboard graphics to produce striking pictures. I won't describe which key does what, since that would ruin the fun of discovery and experimentation.

Here's something to try. Hold down any number key to make a row of moving images. Press the return key to move the cursor

to the left margin on the next line. Type your name and press the return key again. Now use another number key to make a bottom row to highlight your name. The English pound key stops the program. Simultaneously pressing the shift and CLR/home keys clears the screen.

To save a screen, press the f1 key, give the computer the name of your picture and press the return key. Any name will work, provided there isn't another picture of the same name on the disk. The disk drive's red light will come on, and the screen will stop pulsing. When the pulsing starts again, the screen will be saved.

Loading a picture from disk is almost like saving one, except you use the f7 key instead of f1.

When saving or loading, you'll lose the character in the upper left-hand corner when the cursor returns to the home position after disk access. R

Auto-Run

By Alejandro Kapauan

RUN It Right

C-64; C-128 (in C-64 mode); VIC-20
1541 disk drive

Some professional software houses feature programs that automatically run when they are loaded into the computer. This makes the programs easier to use and protects them from being listed and examined.

Now, you, too, can make your programs self-running with Auto-Run, which allows them to begin execution the moment you load them in.

Self-running programs must be loaded with the "1" command of the Load statement, as in LOAD"PROGRAM",8,1. After the program is loaded, it starts execution.

If you load the program without the "1" command, it will not automatically start execution; however, you can neither list nor save the program to produce a working copy.

Self-running programs are possible due to various operating system subroutine vectors in low memory. You can change these vectors to point to machine language routines. Auto-Run creates a program file that actually loads right over these vectors.

One important vector, the warm-start link at location $0302–$0303, is changed to point to a small machine language routine in the cassette buffer. After the program loads, this new warm-start routine issues a Run command to Basic, and thus starts execution of the program.

PROGRAM DETAILS

Auto-Run starts off by printing a header on the screen and Poking the auto-start machine language routine into the cassette buffer. The sum of the data in the Data statements is checked against a checksum to ensure accuracy.

The program asks for the filename of the program for which an auto-start copy is to be made. The program file is then opened for reading.

You are then asked for the name to be given the output file,

that is, the auto-start version of the program. The output file is then opened for writing.

USING AUTO-RUN

To use Auto-Run, first load it into your machine and run it. Then, insert the disk that contains your program into the disk drive. Type in the name of your program. When you are asked for the output name, type in the name you want to give the auto-start copy of your program.

The screen will clear and the program will start working. The program takes a couple of minutes to create the output file—longer for long programs. When the program is done, you can test your new program file.

Type in

```
LOAD"progname",8,1
```

where progname is the name of the auto-start copy of your program. Immediately after loading, the program should start running.

To copy an auto-starting program created by Auto-Run to another disk, you may use any one of various file-copying programs or machine language monitors. You cannot load the program and save it in the normal manner.

If you cannot copy such a file, you can always copy your original program to the disk, run Auto-Run to produce an auto-starting copy, then delete the copy of the original file.

Be sure to remove any cartridges or expansion boards before you run Auto-Run if you expect the auto-start program to run in an environment without them. Or leave them in place if you expect the program to run with them.

If your intention in producing an auto-start program is to prevent your program from being examined, you must take the necessary steps to make your original program airtight. That is, there should be no way of getting out of the program without its consent, short of powering down the machine. Of course, no protection scheme is perfect, so there is always some way of examining any program.

Some simple measures you can take are to disable the run/stop and run/stop and restore keys and to use the New command in the program to delete the program from memory.

To disable the keys, simply type POKE808,234 (normal value is 237) on the C-64 or POKE808,100 (normal value is 112) on the VIC-20. Type these Pokes before you run Auto-Run, and they will carry over to the auto-starting program. A nice side-effect of using these Pokes is that the List function will also be disabled. [R]

Add Elegance to Your Program Listings

By Michael Broussard

RUN It Right

C-64; C-128 (in C-64 mode)

When you're trying to debug or to modify a Basic program, which occurs first—eyestrain or madness?

In Basic, difficult-to-read program listings often result when you crowd statements into program lines to conserve memory and increase execution speed. Under such circumstances, it's not much fun trying to decipher, say, where a loop starts and stops. The accompanying utility, Lister, puts uniform spaces between words, numbers and symbols in your program listings.

Lister is a machine language program that resides in upper RAM. When you run it, it produces a formatted listing of the Basic program currently in memory, making it much more readable.

To use Lister, you must first load it into RAM using the Basic loader, which I'll describe a little later. To run it, you type:

```
SYS 49152
```

Lister first clears the screen, then asks:

```
OUTPUT TO SCREEN OR PRINTER (S/P)?
```

at which time you type either S or P. If you choose the printer, the program automatically uses device number 4. If you are using a different device number, you can customize Lister by changing the value in line 100 of the Basic loader program. If you choose the printer as the output device, Lister next prompts:

```
TYPE IN A HEADING TO APPEAR AT THE TOP OF EACH PAGE:
```

At this point, you type a string up to 60 characters long and

```
    LISTER PROGRAM
    ------------------------------------------------

111 FOR T = 1 TO 200 :
      FOR D = 1 TO 70 :
        PRINT T :
      NEXT D :
    NEXT T
211 FOR S = 1 TO 300 :
      FOR Y = 1 TO 90 :
        PRINT S :
      NEXT Y :
    NEXT S
311 FOR X = 300 TO 1 :
      FOR E = 80 TO 90 :
        PRINT X :
      NEXT E :
    NEXT X
411 PRINT T :
    PRINT X :
```

press the return key. This string, along with a page number, will appear at the top of each page of the output listing. Lister automatically leaves margins at the top and bottom of each page. If you don't want a heading, simply press the return key without typing in a string. The page numbers and margins will still be printed. To suppress the page numbers and the spacing between pages, simply type the @ character in response to the heading prompt.

At this point, Lister works its magic and produces a neatly formatted listing on either the printer or the screen. Spaces are placed before and after Basic keywords and arithmetic operators. Only one Basic statement is printed per line, making each statement easy to read and allowing room in the right margin for notes or corrections on hard-copy listings.

When a For statement is encountered, all the following statements up to the corresponding Next statement are indented two extra spaces. This lets you see at a glance which statements are part of which loops and how the loops are nested.

During the listing display, you can pause the output by pressing any key. Pressing any other key restarts the listing. (This feature is particularly convenient when you're listing to the screen.) If you press the Q key (for quit) while the listing is paused, the program will print out the rest of the current program line, then return immediately to Basic.

The Basic loader Pokes the Lister program into a safe part of RAM, where it won't be disturbed by Basic. You can then load in the Basic program on which you're working. As long as you don't turn off the computer, Lister will be there, ready to produce elegant output whenever you type SYS 49152. R

New Names for Old Disks

By Robert Dickow

RUN It Right

C-64; C-128 (in C-64 mode)
1541 disk drive

The 1541's disk operating system (DOS) provides a fairly comprehensive set of commands. However, it doesn't include a command for renaming a disk, at least not without destroying all the files on it.

Fortunately, the accompanying disk utility remedies this problem, letting you rename your disks as often as you like without damaging them.

When you run the program, it will wait for you to place a disk in the drive, then prompt you to enter the new disk name, which can be up to 16 characters long. After you've done this, you may press the return key to verify the change or insert another disk and repeat the process.

If at some point you decide not to make the change, hit the left-arrow key. The only way of editing your new name is via the INST/DEL key. In any case, all the necessary user information is displayed on the screen after you run the program. When testing this program, I suggest you use an old scratch disk.

To see how the program works, study the commands in lines 230 and 410 to 440. You can find a description of these commands in your disk drive owner's manual in the section on random-access files. A number of these commands can be used to access any byte on the disk, and reading bytes is as easy as writing them.

Following is a description of how to read a byte from the disk. First, you must open the command channel with the statement:

```
OPEN 15,8,15
```

Command channel 15 is needed for locating the bytes you want later on and for reading the I/O errors. A buffer is then set up for a block of data. Open a channel to access the data:

```
OPEN fl, dv, ch, "#"
```

The fl is the file number; dv is the device number (8, usually); and ch is the channel number (any number from 2 to 14). Now, pull the block of data you want into the drive RAM buffer with:

```
PRINT#15, "UA:";dv;0;tr;bl
```

This last command works the same as the User1, or U1, command, and it's similar to the Block Read command. The tr and bl stand for the track and sector you want to access. The dv is, of course, the device number, normally 8.

Finally, point to the byte you want to retrieve with the Buffer Pointer command as follows:

```
PRINT#15, "B-P:";ch;x
```

The x is the number of the byte (1 to 256) that will start the byte-retrieving process, and ch is the channel number used when the data channel is first opened. To recover the disk name, you need to look in track 18, sector 0. This block houses the block-availability map, or BAM. The disk name follows immediately after the BAM data.

There, starting with the 144th byte of track 18, sector 0, and for 16 bytes following, you can find the name. The lower locations in the BAM are important, as they indicate to the DOS where free blocks for saving files are located. Therefore, you must be careful not to write data there accidentally!

From this point on, just use the Get# command to retrieve each successive byte of data. You can then display the data on the screen, transfer it to another file or perform any number of other operations.

The process of writing data to the disk is similar to the process of reading data from it. When reading, however, use the USER2 (or U2 or UB) command after outputting to the datafile via the Print# command. Finally, close the data channel and then the command channel.

The commands discussed in this article can open up a new world of disk exploration for you. Try writing a program that will access blocks of data from the disk drive and allow the user to edit any byte. You can also change the loading address of machine language programs, change sequential files to program files (and vice versa), and even change filenames. Enjoy. R

Ultra Hi-Res Graphics— A Breakthrough on Your C-128

By Louis R. Wallace and David P. Darus

RUN It Right

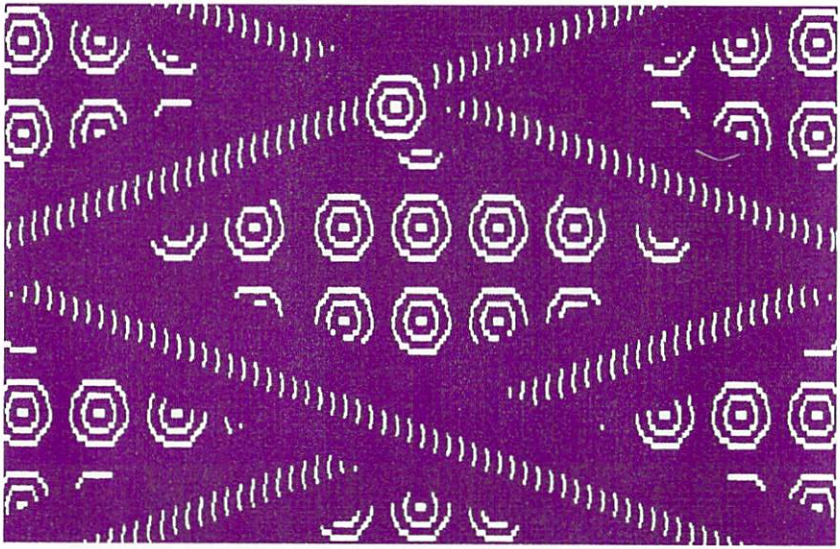
C-128 (in C-128 mode); disk drive

The Commodore 128 computer is a remarkable machine. It contains one of the best Basic languages ever included with a microcomputer, has 128K of RAM and is expandable. It supports high-resolution graphics in Composite mode. It contains both a working C-64 and a Z-80 CP/M computer, has a great keyboard and a beautiful 80-column display. Perhaps best of all, it's inexpensive. What more could you want?

The 80-column screen on the C-128 is also a welcome blessing to those of us who find 40 columns a limitation. This new 80-column capability is made possible by a second graphics chip, the 8563. This is a very powerful device, making possible the crisp, clear, color text on the C-128 in 80-column mode. This chip provides 640 x 200 resolution instead of the standard 320 x 200 resolution that is available in 40-column mode.

There's a lot of mystery surrounding the C-128's graphics capabilities, and many of you must have a lot of questions. This article is meant to dispel them.

First and foremost, the question arises about bit-mapping, or hi-res graphics. Since the 80-column text is made possible by a resolution of 640 x 200, many of



us hoped that we'd be able to use that for graphics. However, Basic 7.0 contains no provision for high-resolution graphics using the 80-column display.

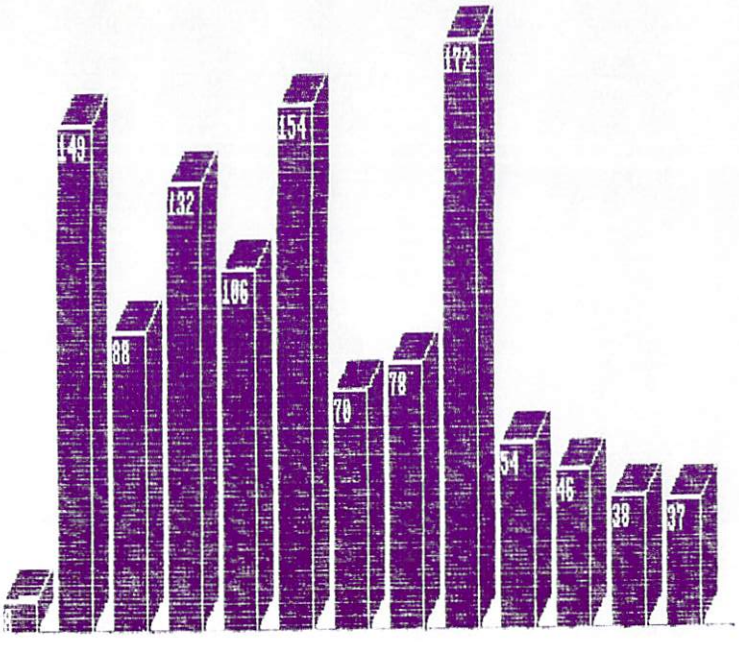
We were told originally that we might be able to do some elementary graphics via custom characters on the 80-column screen, but not true hi-res graphics.

Also, the technical specifications for the 8563 state that it has only limited bit-mapping capabilities. This, more than anything, fired our interest in seeing just how limited it actually is.

GOOD NEWS

We are pleased to announce that our research has led to the discovery that *yes*, it is possible to use the 640 x 200 screen in Bit Map mode! This means that in Composite mode, you can have twice the resolution of the C-64 or C-128. And since it is an RGB display, the graphics are crisper and cleaner than ever. In case you're interested, 640 x 200 is equivalent to the resolution available on the IBM PC and the Amiga (in Non-Interlaced mode).

The 8563 has a special 16K RAM area completely separated from the normal system RAM, so no user RAM is needed for the display. However, because of this, trying to communicate with the chip is very difficult. In fact, there are only two addresses in the whole C-128 memory map that have any effect on the chip



at all. These registers, $D600 (54784) and $D601 (54785), serve as a window from which you may address the chip registers and its 16K RAM bank.

The first address, $D600, is the 8563 register-select byte; the second, $D601, is the 8563 data I/O byte. To access the chip, you must put into $D600 the chip register number you want to read or write to, wait until the chip is ready, and then Peek or Poke into $D601 the value you want. It is quite a bottleneck, effectively precluding any usable hi-res graphics using Peeks or Pokes from Basic. But from machine language it's a different story.

We have created a set of machine language commands that are wedged into the C-128 to take advantage of its 640 x 200 graphics power. These will give you a very powerful command set that works with Basic 7.0 and allows you to use these new graphics freely.

There are 14 new commands (see Table 1), most of which are used to work with the 640 x 200 display. One, @FONT, is a special

command—for 80-column Text mode only—that allows you to download new character sets from system RAM to the 80-column chip RAM. Since the 80-column text screen allows for two complete character fonts at once, you can use @FONT for some very interesting text displays.

ACCOMPANYING PROGRAMS

Included with this article are several programs.

The first program is called Ultra Hi-Res Boot and is used to load the Ultra Hi-Res machine language module and set up the computer for its use. You should run it only once, as Ultra Hi-Res creates some tables that cannot be written over. The best method is to use the C-128's autoboot feature and create an autoboot disk that loads and runs Ultra Hi-Res Boot for you. This boot performs several functions. First, it issues a

```
GRAPHIC 1,1:GRAPHIC 5
```

command. This is the only time you should use the normal Graphic command. It allocates a 9K area in the computer to be used for composite hi-res graphics. Since you're going to be using the new Ultra Hi-Res mode, this area can now be used to hold the machine language commands in the wedge.

Next, it Pokes the start-of-Basic variables in bank 1 up 16K, giving you a 16K RAM buffer for the @STASH command.

It then loads the Ultra Hi-Res machine language module and activates it with SYS 8448. At this point, you now have the wedge active.

The second program is called Ultra Hi-Res. It is a machine language module that you can easily load into memory using the boot program.

The third program is called Program Base and is intended as a base from which you can start writing your own programs.

It begins by going into Fast mode. One of the nice features about the 80-column chip is that you can use the Fast command to allow the computer to run at 2 MHz instead of 1 MHz, and the screen display is always visible. (In Composite mode, the screen goes blank during Fast mode.)

Next, it issues the Poke to set up the 16K buffer for @Stash (see @Stash later in this article for information about increasing or decreasing the buffer). The new @Graphic command is issued next, going to Ultra Hi-Res mode and setting up background and foreground colors.

Line 50 issues the @CLR command with a value of 0, which will clear the 640 x 200 bit-map display, and line 60 sets up the Basic 7.0 Trap command. This is very

important because, in case of a Syntax error or program crashing, it will instruct the computer to go to the line following the Trap instruction. Here it goes to line 10010, which gets you out of Ultra Hi-Res mode, prints the line that has the problem (with the Help command) and ends the program. Even pressing the stop key is handled by the Trap statement. It is highly recommended that you include the Trap statement in your programs.

THE DEMO PROGRAMS

Next are five short demo programs that use various forms of the new command set. The first is an example of the versatility of the @CHAR command, which allows many different sizes of text (and many different styles) all at once. It even has a special form that will give you 160 columns on one line! Imagine the possibilities!

The second demo program is a simple line-drawing demo that creates a very interesting graphics effect called a *moire* pattern. It looks very good in 640 x 200.

The third demo is one that uses the 3-D Bar command. This command allows you to create three-dimensional bar graphs so easily it will amaze you (and your friends)!

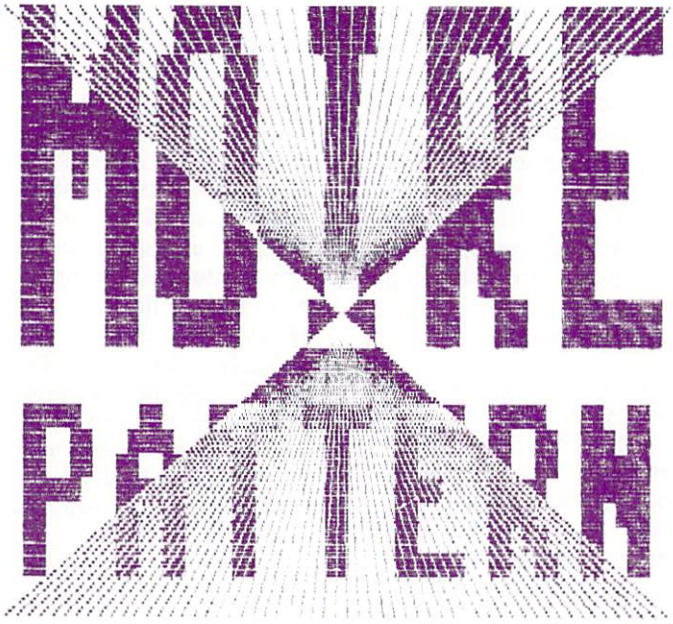
The fourth demo uses Basic to create circles, and the circle routine could very easily be modified for use as a general-purpose subroutine. After the circles are drawn it uses the @Copy command to duplicate them, and then uses the @Stash and @Fetch commands to create a form of animation called *blitters*. That's right, the C-128 now has blitter graphics capabilities. (Blitters are a form of high-speed bitmap transfers, often mentioned in reference to Commodore's new Amiga computer.)

The last demo is an impressive demonstration of the power of the new commands. It creates a fully animated demonstration of a three-dimensional cube rotating and changing size in real time. It uses the @Stash and @Fetch commands (and a 32K @Stash buffer) to create a small movie. After seeing this, you'll probably be highly motivated to begin using these new Basic commands on your C-128.

THE C-128'S MEMORY

The memory organization of the C-128 is a very complex subject, and we cannot go into it in detail in this article. However, you need some information to use the Ultra Hi-Res command set effectively.

In an unexpanded C-128 there are two banks of RAM, bank 0 and bank 1. Bank 0 is where your Basic program resides, and bank 1 is where your variables



are stored. When you enable the normal composite bit-map graphics display with the

GRAPHIC 1,1

statement, the C-128 moves your program in bank 0 up 9K to make room for the graphics screen and its color memory. That's why the Ultra Hi-Res Boot program issues it as its first statement.

This area can now be used to store the machine language program that gives you Ultra Hi-Res mode. If you were to use that area again in a program while in Ultra Hi-Res mode, it would erase the whole program and crash the computer.

This is also true of graphic multicolor composite Bit Map mode, which you should never use in your programs while Ultra Hi-Res is enabled. The same thing is true of the Graphic CLR command, which de-allocates the 9K, making it available for Basic. And the SCNCLR command should never be used, for the same reasons as the Graphic command.

Bank 1 contains about 64K to be used for variables. Registers 47 and 48 contain the address for the start-of-Basic variables. The normal value for 47 is 0; for 48, it's 4 (this is the address $0400 in hex and 1024 in dec-

imal). Everything above that is used for variables. But by Poking a larger value in register 48, followed by the Basic CLR command, you can trick the C-128 into giving you some extra RAM that you can use in your programs. The Ultra Hi-Res Boot program, for example, automatically sets up an extra 16K buffer in line 20 with

```
POKE47,0:POKE48,68:CLR
```

You can increase or decrease this buffer by changing the contents of register 48. (Be careful not to make it less than 4!) If your program will be using a lot of variables and won't be using the @Stash and @Fetch commands, you won't need a buffer area. In this case, you might want that RAM to be used strictly for variables, so start off your program base with the line:

```
POKE47,0:POKE48,4:CLR
```

However, some programs (such as the 3-D cube demo) require even more RAM in their buffers, so you will have to increase the size by Poking to registers 47 and 48.

```
POKE47,0:POKE48,132:CLR
```

This provides you with a 32K buffer in which to store graphics. However, you've now cut your variable RAM in half. You will have to decide if you need to change it. In most cases, the default of 16K will be sufficient.

One more point. The screen resolution (640 × 200) calculates to 128,000 bits, or 16K of RAM. That's exactly the amount of RAM available to the 8563 chip, and does not leave any RAM for 80-column Text mode. If you need to use the Text mode and have an important screen in Hi-Res mode, you will have to save that screen (to disk, using the @Save command, or in the @Stash buffer), go to Text mode with the @Text command, perform your needed function, return to Ultra Hi-Res mode with the @Graphic command and restore your screen from the buffer or disk.

When would you need to do this? One time might be when you need to input some value from the keyboard while the program is running. The Input command prints a question mark on the text screen. In Ultra Hi-Res mode, however, there's no RAM left for the 80-column text screen, so the operating system would corrupt your Ultra Hi-Res program with the question mark. Therefore, use the GETKEY in place of the Input command, unless you toggle back to Text mode.

C-128 ULTRA HI-RES COMMANDS

After you have successfully activated the C-128 Ultra Hi-Res program, the following com-

mands are available to you in addition to all of Basic 7.0 (except those graphics previously mentioned).

@FONT,character set #,RAM Address. This command allows you to display (in 80-column Text mode) character fonts other than those offered by the default character set. There is room in the 8563 RAM for two sets, and they are normally the uppercase/graphics and the lowercase/uppercase sets. If you have access to other fonts that you wish to use, simply load them using the BLoad command, to some area of RAM in bank 0 (in Direct or Program mode) and issue the @Font command. There is 3K available for this, from 12992 to 16383. The Char Set # is either 0 or 1, and the RAM address is where BLoad placed the character set. You can change character fonts by pressing the shift and Commodore keys simultaneously or by printing CHR$(14) or CHR$(142).

@GRAPHIC,BC,FC. This turns on the 640 x 200 Bit Map mode. BC is background color and FC is foreground color. When you're in Ultra Hi-Res mode, you are limited to two colors, and this command lets you choose them.

@TEXT. This turns off the Ultra Hi-Res mode and returns you to the normal 80-column text screen. You also return to the standard character font, so you'll have to issue the @Font command to re-enable any extra text fonts you want.

@CLR,value. This is used to clear the Ultra Hi-Res screen. Use a value of 0 to clear it and a value of 255 to fill it. Others can be used for special effects.

@DOT,X,Y,mode. This is used to plot a dot on the 640 x 200 screen. X is from 0–639; Y is from 0–199. Mode is either 0 (for erase) or 1 (for draw).

@DRAW,X1,Y1,X2,Y2,mode This is the line-drawing command. X is 0–639; Y is 0–199. Mode is the same as in @DOT.

@BOX,X1,Y1,X2,Y2,mode. This will draw a box on the 640 x 200 screen. X1,Y1 are the coordinates of the upper-left corner; X2,Y2 are those of the lower-right corner. X is 0–639 and Y is 0–199. Mode is as in @DOT.

@BAR,X,Y,dX,dY,Ht,mode. @BAR draws a 3-D bar of a given height. X and Y are the coordinates of the lower-left side of the bar; dX and dY are the depth and width you wish; Ht is the height; mode is as in @DOT; X is 0–639; Y is 0–199; dX is 1–255; dY is 1–199; Ht is 1–199. The sum of Ht and dY cannot exceed 199; if it does, no bar will be drawn.

@SAVE,type,"filename". This will save to disk your hi-res screen display (not your program listing) called "filename." There are two types of saves. Type 0

is a normal 16K screen dump, which will give a disk file of 65 blocks. Type 1 is a special compressed form of save, where an intelligent data-compression process will shrink your screen to its smallest possible size. In some cases, it can cut the 65 disk blocks down to only a couple of blocks; but in other cases, it can cut more than 50% off the size of your file! The actual amount of reduction will depend on what is being displayed. It will never be bigger than 65 blocks. In most cases, this Compressed mode will be the best way to save your screens. Use the regular Basic 7.0 Save and Load commands for saving and retrieving your Ultra Hi-Res program listings.

@LOAD,type,"filename". This loads a screen called "filename" from disk. Type is either 0 or 1, depending on how it was saved. Be sure to load your file under the same type, or you will get a scrambled screen. Use the @Load and @Save commands in Program mode.

@CHAR,charset address,X,Y,Ht, Wd,"string". This is the high-resolution character driver. It allows you to print on the 640 × 200 bit-map screen in many sizes and styles. X is any number 0–639; Y is 0–199; Ht is 1–16; Wd is 0–16; and string is what you want it to print, either in quotes or as a string variable. The address is where in RAM you want the character set to be taken from. You can use the built-in sets at 53248 and 55296, or, with BLoad, you can load others into RAM and use them.

Built into the machine language is a special font that allows 160 characters per line. Its address is at 7168, and it requires a width of 0. You can use almost any character set made for the C-64 or C-128, as long as they are binary files. Check your user's group library for extra fonts, as it will probably have many.

You may also use special control codes inside the string, such as reverse-on and reverse-off. Control E causes the text to erase anything under it, while Control X performs an XOR on the screen, leaving anything already on your screen still visible. You can also underline your text with Control U and turn off underlining with Control N. Color codes have no effect.

@COPY,SX,SY,dX,dY,destination X,destination Y. This will allow you to duplicate any rectangular area of the screen in any other area. SX,SY are the starting coordinates of the upper-left corner you wish to duplicate. The dX and dY are the lengths (in pixels) you want to copy from SX,SY; dX is from 1–640, and dY is from 1–200. Destinations X and Y are the coordinates of the upper-left corner of the new position.

@STASH,buffer address,X,Y, dX,dY. This command allows you to store to the buffer a section of the screen and then recall it when needed. X and Y are the upper-left coordinates of the screen you want to save, and dX and dY are the lengths. The section of screen you store may be as small as a byte or as large as a screen. The buffer address is the location in the buffer where the piece of screen is to be stored. The buffer starts at 0, so the first section of screen you store should go to that address. You will need to know where the first stash of data ends in the buffer so you can store other screen data after it. You can find the next available buffer address with:

```
AD = PEEK(250) + PEEK(251)*256 + 1
```

You must do this immediately after issuing the Stash command. AD now contains the next available buffer address. You must keep these addresses stored so that you can recall them later with the @Fetch command.

@FETCH, buffer address,X,Y. This will recall the stored area at the buffer address and put it at X,Y. No other information is needed, as the @Stash command saves the length and depth of the area. The @Stash and @Fetch commands are very rapid, fast enough for some types of animation. See the circle demo and 3-D cube demo programs. You can use a simple For. . .Next loop with @Fetch to accomplish some pretty impressive blitter animation.

ROOM FOR EXPANSION

The 8563 chip can perform many unsuspected functions. It can scroll and handle light pens, double-pixel modes, interlaced modes and more. We've only begun to explore it, and the wedge is written to make it easily expandable to handle new commands (perhaps a Hard Copy command or a Fill). There's 9K of RAM set aside for growth; also, the program only goes from 7168 to 12992. That leaves room for you to add even more commands in the area from 12992 to 16383. For now, you could use that free RAM for character-set storage, but be careful and don't disturb the RAM from 7168 to 12992.

I'd like to thank Andy Finkle and Carolyn Sheppner of Commodore for their kind assistance on the 8563's technical aspects. R

Tax Deductor's Delight

By Barbara Schulak

RUN It Right

Commodore 64; Disk or tape
Printer optional

Itemized Tax Deductions will keep a file of all your itemized deductions. The fields for each record are category, date, check number, description and amount. The categories are—as on Schedule A—medical and dental expenses, taxes, interest expense, contributions, casualty and theft and miscellaneous.

Records may be added, changed or deleted. They may also be displayed on the screen or outputted to the printer, either as a whole or by category. The latter option provides a sort of all the records by category and includes a statement of each category's total expenditures. The data may then be saved to either disk or tape. The program allows for the filing of 200 individual records. If more are required, the DIM statements in line 5 should be changed.

The program first presents a main menu. The options are as follows:

1. Load data
2. Add a record
3. Find a record
4. Change a record
5. Delete a record
6. Display file
7. Print file
8. Save data
9. End program

When running the program for the first time, choose Option 2, add a record. After the first session and once data has been saved, you must start with Option 1, load data.

The program will then ask for a filename. I would suggest using something like TAX-1985. If you're using a disk, the program will display a message that gives the status of the load. Then the program returns you to the main menu.

To add data, choose Option 2. The prompts are category, date, check number, description and amount of check. The category designation is entered numerically (1–6) from the displayed list. To enter the date, you may use any of the following types of formats: AUG 01, AUG 1, 08/01 or 8/1. Mainly, you should be consistent and not use more than six characters (including spaces).

The check number is entered as a string, so you may enter other strings of four characters or less (such as CASH) if the expense was not in the form of a check. The description can be anything, as long as it's 20 characters or less. Do not use dollar signs ($) when entering the amount. Enter 100 or 100.00, but *not* $100. Again, consistency will provide you with neater outputs.

In addition to adding records to the file, you may also change or delete an entry. To find which entry to change, you may use Option 3, find a record, which will sort through the records by any category or field. Or you may use Option 6, display file. Know your check number, because it is the key field for deleting or changing an entry. Then, just follow the prompts on the screen, using the same guidelines for data input as outlined above.

To see your file, choose Option 6. A submenu will then ask if you want to see all the records, or records sorted by field. You may then choose the field that you want. Any of these options results in the output of all necessary information to the screen, including a total of the amount spent.

Option 7, printing the file, works in the exact same manner.

Option 8 is the Save File routine and is exactly like the Load routine. Don't forget to save the data before ending the program (Option 9)!

This program might not do everything for you in Schedule A, but it is an excellent way to collect, store and sort your records for tax purposes and should also save you some time. R

Fast-Food Chef

By George Decker

RUN It Right

C-64; C-128 (in C-64 mode)

Burgers is a game that will drive you a little crazy. In these days of mass production, fixing a burger shouldn't be at all difficult. Push a computer key and on squirts the catsup. Push another key and you've got the mustard. Still another finishes the burger by putting the top bun in place. Sound easy? Well, what if the keys you needed to push kept changing? And suppose the burgers kept coming faster and faster? You might get a little frazzled.

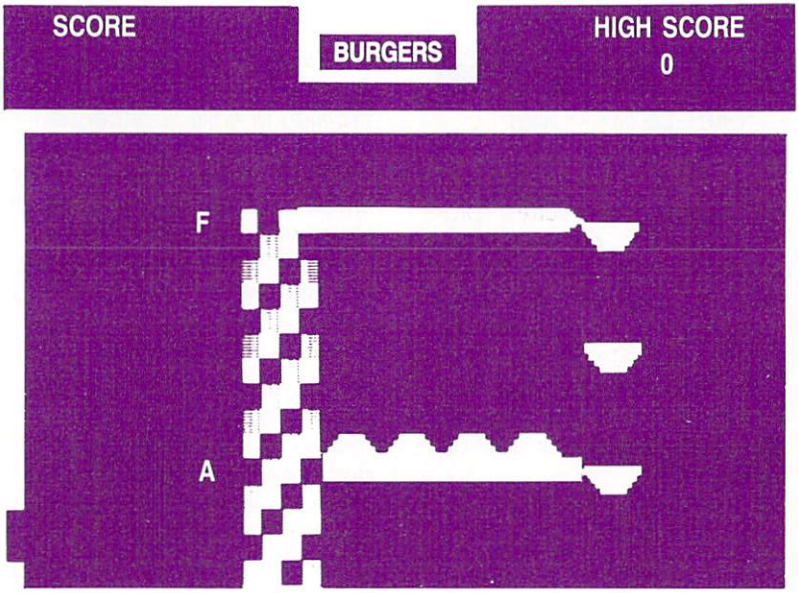
Burgers begins with the bottom bun moving down the screen. It will pause (the duration depends on the skill level you select) beside the catsup, mustard and top bun holder. Next to each item will be a letter that changes as the bottom bun moves. When the bun pauses, the letters will stop, and you must type in the three letters selected.

You must build each burger in a specific order: catsup first, mustard second and top bun last. If you miss putting on the catsup, then you can't add the mustard and top bun. To add the bun, you must top the burger with both catsup and mustard.

At the bottom of the screen is a red bar showing what fixings you must add to the burger. Under this is your score. Every unfinished burger causes the red bar to go down, and if this bar runs out, the game is over.

You receive 100 points for every finished burger. For every 1000 points you earn, the fixings bar will rise. High score is displayed at the top of the screen.

There are three skill levels: easy, normal and hard. Each level determines the length of the bun's pause. As you play, regardless of which level you select, the pauses will get shorter and shorter. R


SCORE
BURGERS
HIGH SCORE
0
F
A

Datafile Restructure Utility

By Mike Konshak

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive

The Datafile Restructure program, hereafter called DFRestructure, will allow you to safely change the structure of existing files created by Datafile (*RUN*, November and December 1984 and ReRUN Productivity Pak 1985) and to save the new file under the same name or a different one.

The following operations are possible with DFRestructure:

- Change the name or title of a field.
- Change the length of a field.
- Add an additional field.
- Delete an existing field.

Since Datafile is a memory-based, rather than a disk-based, system, storing the records on disk in sequential files, you can alter the field structure—add or delete fields—without destroying all your data.

The number of records that may be held in memory after the creation of a file is dependent on the size and number of the fields within a record. Consequently, when you change the datafile structure (except for a field name change), the number of possible records will be altered. The operations that affect the number of possible records are as follows:

- Changing the length of a field—Increasing a field's length or the number of characters within it will decrease the number of possible records. Decreasing the length will increase the number.
- Adding a new field to the structure—This obviously increases the size of each record, since an additional number of characters must be allocated for the new field. This results in less space available for records in memory. However, if you need additional information that is not specified in other fields, this sacrifice of space may be necessary.
- Deleting an existing field—This

frees up more memory space, because each record is reduced by the number of characters found in the deleted field. You might choose this feature to eradicate a field you no longer consider important.

Before DFRestructure makes any changes to your datafile, it will recalculate the number of possible records based upon your selection. You will always be given the chance to change your mind before the restructured file is saved to the disk. If the file with the new structure is given a new name, then the original file will be untouched. If the original name is chosen, then the original file will be renamed with !OLD at the end of the filename. This allows you one more chance to recapture that particular datafile and structure.

The restructuring actually occurs during the writing, or saving, of the new file to the disk. As the program reads the structure information, the drive will operate at various times throughout the procedure.

Only one of the four options may be performed at a time, with the modifications being saved to the disk after each process. If the same file is used throughout the operations, you won't have to restart the program. You must start at the beginning if you wish to modify a different file.

The DFRestructure program is really very straightforward and self-explanatory. It uses many prompts and describes the operations when necessary. The structure of the datafile will always be displayed before and after modifications are made, so you'll see the result. If you really want to be cautious, save the new datafile structure on a disk separate from the original. This will further protect your originals.

You may view the final results of your efforts by loading Datafile and reading your new restructured file. It is at this time that you might consider entering the Modify All Records option (if you added an additional field) in order to enter the data that prompted the change. R

Datafile Structure Utility

By Mike Konshak

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive; printer optional

The Datafile Structure utility, or DFStructure, allows you to peek at the sequential files created by Datafile or other programs. DFStructure reads the files generated by the following programs and produces a printout, on either screen or printer.

These programs are: Datafile (database program; *RUN*, November 1984); DFReport and DFMail (both sub-programs of Datafile; December 1984); and DFCalc (companion program to Datafile; in the recent ReRUN Productivity Pak). The last three programs produce format files for printing out records that Datafile creates.

Each program in the Datafile series writes sequential files to the disk using a unique code that precedes the filename. It is possible for all these programs to create files with the same name. The difference, which becomes apparent only when you view the disk directory, is in the special characters that precede the filenames.

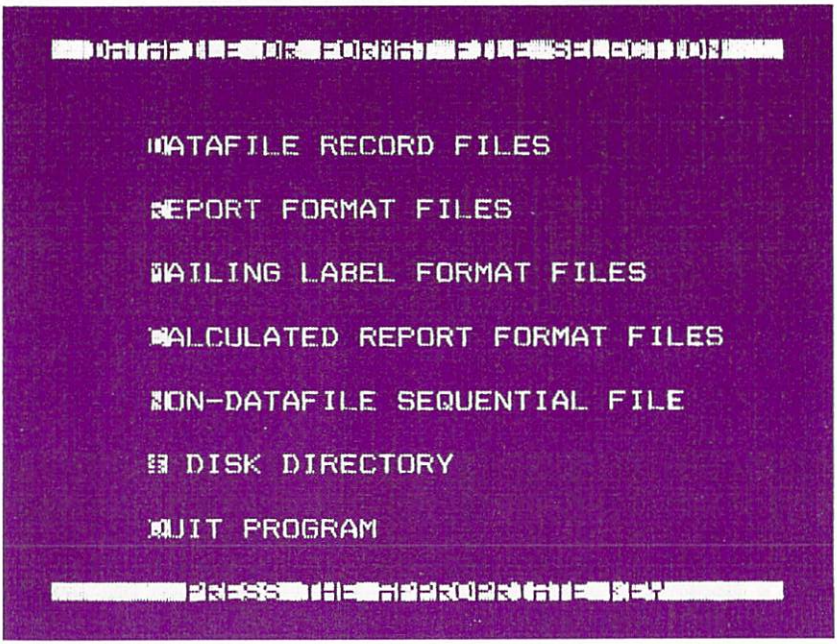
The following codes, which precede the filename, SAMPLE, are written by the respective programs:

```
DATAFILE . . . . .DF]   SAMPLE FILE
DFREPORT . . . .RP]   SAMPLE FILE
DFMAIL . . . . . .ML]   SAMPLE FILE
DFCALC . . . . . .CR]   SAMPLE FILE
```

Every sequential file in the Commodore operating system is further identified on the directory with a SEQ after the filename.

BEGINNING DFSTRUCTURE

To use DFStructure, first type LOAD"DFSTRUCTURE",8 and press the return key. Then type RUN <return>. You'll first see an introduction screen, followed by this menu:



[DATAFILE OR FORMAT FILE SELECTION]

[D]ATAFILE RECORD FILES
[R]EPORT FORMAT FILES
[M]AILING LABEL FORMAT FILES
[C]ALCULATED REPORT FORMAT FILES
[N]ON-DATAFILE SEQUENTIAL FILES
[$] DISK DIRECTORY
[Q]UIT PROGRAM

[PRESS THE APPROPRIATE KEY]

(Letters or words surrounded by brackets denote reversed printing on the screen and normally identify a prompted key to be pressed. Letters or a word surrounded by inequality signs denote an actual key to press.)

Pressing $ will display the directory of the disk currently in the drive. When you view the directory of files created by Datafile, you'll notice that the filenames are preceded by two characters and a right bracket (i.e., DF]. . .). These special codes allow the use of identical names for files, yet keep the names distinct for the disk operating system.

Pressing D, R, M, C or N will

initiate a prompt for the name of your sequential file. If the file is one of the Datafile programs, just enter the name without the special characters. If not, you must type in the name exactly as shown on the disk directory.

After you enter the name and press the return key, the program will search out the data from the disk and display it. Different files use different routines, but they are all very simple to understand.

If, for example, you want to see the format of the datafile MAIL LIST, you press the D key to view the structure of the file. You'll be asked for the name of the file. If you had previously entered a filename, it would automatically be printed at this point. Do *not* use the special codes as they appear on the directory, but enter the name as shown below.

```
ENTER NAME OF SEQUENTIAL FILE:

? MAIL LIST     <Return>
```

If you enter a filename that is either not on the disk or not a Datafile file, then you will receive a File Not Found error message.

After you enter the filename, the disk drive will start running and the screen will display the following (assuming you are reading the format of the sample file MAIL LIST):

```
[STRUCTURE OF DATAFILE MAIL LIST]

# RECORDS POSSIBLE IN FILE [192]
# RECORDS IN CURRENT FILE [4]
# FIELDS IN EACH RECORD   [8]

[#   TITLE OF FIELD   LENGTH]
[1]  LAST NAME..... 15
[2]  FIRST NAME ..... 10
[3]  CODE ..... 5
[4]  STREET ..... 32
[5]  CITY ..... 23
[6]  STATE..... 2
[7]  ZIP ..... 5
[8]  PHONE..... 12

[S]CAN RECORDS
[P]RINT STRUCTURE
[E]XIT
```

You'll notice that the red light on the disk drive is still on at this time. The file is actually still open, waiting for your next decision.

Pressing E will exit you back to the main menu, closing the file in the process. The red light should then go out.

Pressing P will produce a hard copy of the file structure by sending the data to the printer. The printout will be similar to that shown on the screen. When the printer has finished its operation, you'll be returned to the above screen.

Pressing S will allow you to view in turn the datafile records in the current file structure. Before DFStructure begins this operation, it shows the following:

```
PRESS [CTRL] SLOW [F7] START/STOP
      [F1] EXIT
```

Pressing any key will then start reading and displaying the records in the datafile. The record number will be displayed before each record, followed by the field number and the data within each field.

Pressing <CTRL> will slow down the scanning operation so that you may view the data more easily.

Pressing the f7 key will stop the operation after a record has been printed on the screen, and pressing f7 will again start the scan.

Pressing the f1 key will let you exit the scan and bring you back to the structure screen.

DATAFILE RECORD FILE STRUCTURES

Each of the various files created by the Datafile series will have different information contained within a structure display. As shown above, the Datafile record files contain the following information:

- Number of records possible in a file.
- Number of records in the current file.
- Number of fields within each record.
- Title of each field.
- Length of each field.
- Data contained in each record in fields.
- Index number used for sorting records.

You may view the actual string and numerical data, as they appear on the disk, by selecting the Non-Datafile Sequential File option at the main menu. This will show you the exact sequence of the contents of the datafile as it resides on the disk. The data will be displayed serially, without any format or labels.

If you use this option, the filename must be entered exactly as it appears on the disk directory (i.e., MAIL LIST is actually DF] MAIL LIST on the directory). This option may be used for *any* sequential file that you may have in your library.

DFREPORT FORMAT FILE STRUCTURE

Report format files, created by DFReport and used for the purpose of producing printed reports in custom forms, will contain the following:

- Number of lines in report title.
- Data for each title line.
- Number of columns in the report.
- Position of each column.
- Header data for each column.
- Which datafile fields appear in each column.
- Whether or not totaling of the last column was chosen.

DFMAIL FORMAT FILE STRUCTURE

Label format files, created by DFMail, are used to print Datafile

records on labels. The custom label structure file contains the following:

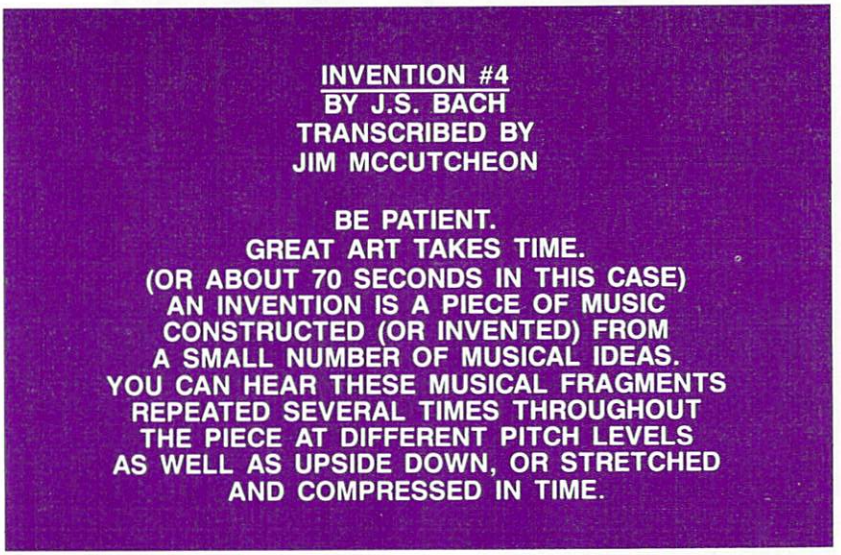
- Number of rows on label.
- Which datafile fields appear in each row.

DFCALC FORMAT FILE STRUCTURE

The calculated reports, simulating the types of reports produced by spreadsheets, are created by DFCalc. These format file structures contain the following data:

- Number of lines in report title.
- Data in each title line.
- Number of columns in report.
- Position of each column.
- Header data for each column.
- Contents of each column (datafile fields, rec# or equation).
- Equation(s) for column, if chosen.
- Justification of each column.
- End-of-column operation (totals, averages, NOOP).

DFStructure is especially useful when dealing with calculated reports, because it allows you to use information found in other report formats to help you design new formats. R



plified. This is important,since the data (see lines 500–632) is usually quite extensive.

HANDLING DATA

Data for each note is typically handled one of two ways. It can be read directly into the program where the values can be Poked immediately into the proper locations, or the data can be read into arrays that store it in a more accessible way.

The advantage of the first way is that as soon as the program is run, the music begins immediately and no memory is used to store the arrays, which can be quite large for longer pieces.

The array format offers the advantage that each voice may be programmed by itself, separated from the others in the Data statements. This helps reduce programming errors and also makes troubleshooting easier. Arrays are also easily accessed in case a section of the music needs to be repeated.

I chose the array format for this piece in order to demonstrate how the loading time could be made more educational. While the arrays load and the program plays, the screen displays four interesting facts about the music. If you choose to hear the music in a different key, the paragraph changes to provide you with new information. This way, you can familiarize yourself with the music while learning its history. R

The Classical C-64

By Jim McCutcheon

RUN It Right

C-64; C-128 (in C-64 mode)
Disk drive or Datassette

The music of Johann Sebastian Bach and his contemporaries has long been a favorite source for musicians who play guitar, saxophone, piano, harmonica and a whole host of instruments that didn't exist when Bach lived. The computer, as we know it today, is also a post-Bach invention, and ever since the release of *Switched On Bach*, a decade ago, people have been making music on their computers.

Bach's music represents the culmination of Baroque (rhymes with Poke) music, which consisted of writing more than one independent melody, or voice, to be played at the same time. Individual notes were called points, and the melodies were played against, or counter to, each other—hence the term counterpoint is used to describe this style.

Contrapuntal music is ideal for the 64—the SID (sound interface device) produces three practically independent voices (they are not independent in amplitude) that work very well when transcribing, or rewriting, Baroque music for the computer.

Making the 64 produce music requires a few Pokes to memory locations that control the waveshape of each musical tone. The attack/decay, sustain/release, volume and filtering are set before the music begins playing. Notes are turned on and off with values Poked into memory locations used for the selection of each voice's waveform. (More complete explanations can be found in the *Commodore 64 Programmer's Reference Guide* and various books and articles.)

Each note requires two final Pokes—one high- and one low-frequency value to determine the pitch of the note. These two values can be calculated from a single decimal number so that typing in the data can be sim-

Please send me back issues of ReRUN

____ Vol. I
____ Vol. II
____ Spring Edition
____ Gamepak
____ Summer Edition
____ Fall Edition
____ Productivity Pak (Disk only)
____ Winter Edition

____ C-64
____ VIC-20

____ Cassette version(s) at $11.47*
____ Disk version(s) at $21.47

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Signature

Name

Address

City State Zip

ReRUN • 80 Pine Street • Peterborough, NH • 03458

* Prices include postage and handling. Foreign Air Mail please add an additional $1.50 per item. U.S. funds drawn on U.S. banks only.

BEAT THE RUSH—

Please send me:

☐ 1 year (6 issues) for $89.97
☐ March/April ReRUN disk for $21.47.*

*Available in April.

Programs run on C-64 and C-128 (in C-64 mode).

Price includes postage & handling. Foreign Air Mail please add $1.50 per item. U.S. funds drawn on U.S. banks only.

☐ Payment Enclosed ☐ MC ☐ VISA ☐ AE

Card # Exp. Date

Signature

Name

Address

City State Zip

ReRUN • 80 Pine Street • Peterborough, NH • 03458

18 RUN *Programs Included on This Disk:*

Business and Home Applications ► Telecommunications ► Utilities ► Games ► Graphics ► Music

From the January RUN:

- ►Auto-RUN 64
- ►Word Jumbler
- ►Pulsing Pictures
- ►Lister 64
- ►Runterm Plus, Part II

From the February RUN:

- ►C-128 Ultra Hi-Res Graphics
- ►Tax Deductor's Delight
- ►Disk Renamer 64
- ►Fast-Food Chef

Plus: Selected programs from the December 1985 RUN:

- ►64 Basic Aid
- ►Stack
- ►Credit Card Keeper
- ►128 Perfect Typist
- ►Runterm Plus, Part I
- ►ML Perfect Typist
- ►Autoboot 128
- ►DF Structure Utility

Also:

- ►Baroque Music on Your C-64
- ►DF Restructure Utility

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

ReRUN • 80 Pine Street • Peterborough, NH 03458

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

The programs in ReRUN are taken directly from listings prepared to accompany articles in *RUN* magazine. They will not run under all system configurations. Use the RUN It Right information included with each article as your guide.

The entire contents are copyrighted 1986 by CW Communications/Peterborough. Unauthorized duplication is a violation of applicable laws.

© Copyright 1986 CW Communications/Peterborough

CW COMMUNICATIONS/PETERBOROUGH